

Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Факультет инфокоммуникационных технологий

Направление подготовки 11.03.02

Лабораторная работа №8

Использование интерфейсов при реализации иерархии классов

Выполнил:

Доценников Никита Андреевич

Группа: К3221

Проверил:

Иванов Сергей Евгеньевич

Санкт-Петербург

2025

Цель работы:

Использовать интерфейсы при реализации иерархии классов как важного элемента объектно-ориентированного программирования и приобретение навыков реализации интерфейсов.

Упражнение 1. Создание и реализация интерфейса.

В этом упражнении я создал интерфейс, определяющий поведение классов, которые будут его реализовывать.

```
using System;

interface IPubs
{
    void Subs();
    bool IfSubs { get; set; }
}
```

```
using System;

class Magazine : Item, IPubs
{
    private String volume;
    private int number;
    private String title;
    private int year;

    public Magazine(String volume, int number, String title,
int year, long invNumber, bool taken) : base(invNumber, taken)
    {
        this.volume = volume;
        this.number = number;
        this.title = title;
        this.year = year;
    }

    public Magazine()
    {}

    public override void Show()
    {
        Console.WriteLine("\nЖурнал:\n Том: {0}\n Номер: {1}\n
```

```

Название: {2}\n Год выпуска: {3}", volume, number, title,
year);
        base.Show();
    }

    public override void Return()
    {
        taken = true;
    }

    public bool IfSubs { get; set; }

    public void Subs()
    {
        Console.WriteLine("Подписка на журнал \"{0}\": {1}.",
title, IfSubs);
    }
}

```

Тесты:

```

public static void Main(string[] args)
{
    Magazine mag1 = new Magazine("О природе", 5, "Земля и мы",
2014, 1235, true);
    mag1.TakeItem();
    mag1.Show();
    mag1.IfSubs = true;
    mag1.Subs();
}

```

Пример:

```
Журнал:
  Том: 0 природе
  Номер: 5
  Название: Земля и мы
  Год выпуска: 2014
  Состояние единицы хранения:
  Инвентарный номер: 1235
  Наличие: False
  Подписка на журнал "Земля и мы": True.
```

```
^ labs/lab8/MyClass ↵ main ? >
```

Упражнение 2. Использование стандартных интерфейсов.

В этом упражнении я применил интерфейс `Comparable`, который задает метод сравнения объектов по принципу больше и меньше, что позволяет переопределить соответствующие операции в рамках класса, наследующего интерфейс `Comparable`.

```
using System;

abstract class Item : Comparable
{
    protected long invNumber;
    protected bool taken;

    public Item(long invNumber, bool taken)
    {
        this.invNumber = invNumber;
        this.taken = taken;
    }

    public Item()
    {
        this.taken = true;
    }
}
```

```

    public bool IsAvailable()
    {
        return taken;
    }

    public long GetInvNumber()
    {
        return invNumber;
    }

    private void Take()
    {
        taken = false;
    }

    public void TakeItem()
    {
        if (this.IsAvailable())
            this.Take();
    }

    abstract public void Return();

    public virtual void Show()
    {
        Console.WriteLine("Состояние единицы хранения:\n
Инвентарный номер: {0}\n Наличие: {1}", invNumber, taken);
    }

    int IComparable.CompareTo(object obj)
    {
        Item it = (Item)obj;
        if (this.invNumber == it.invNumber) return 0;
        else if (this.invNumber > it.invNumber) return 1;
        else return -1;
    }
}

```

Тесты:

```
public static void Main(string[] args)
{
    Item[] items = new Item[4];
    items[0] = b1;
    items[1] = b2;
    items[2] = b3;
    items[3] = mag1;

    Array.Sort(items);

    Console.WriteLine("\nСортировка по инвентарному номеру");
    foreach (Item x in items)
    {
        x.Show();
    }
}
```

Пример:

```

Сортировка по инвентарному номеру

Книга:
  Автор: Пушкин А.С.
  Название: Капитанская дочка
  Год издания: 2012
  123 стр.
  Стоимость аренды: 12
Состояние единицы хранения:
  Инвентарный номер: 0
  Наличие: True

Книга:
  Автор: Лермонтов М.Ю.
  Название: Мцыри
  Год издания: 0
  0 стр.
  Стоимость аренды: 12
Состояние единицы хранения:
  Инвентарный номер: 0
  Наличие: True

Книга:
  Автор: Толстой Л. Н.
  Название: Война и мир
  Год издания: 2013
  1234 стр.
  Стоимость аренды: 12
Состояние единицы хранения:
  Инвентарный номер: 101
  Наличие: False

Журнал:
  Том: 0 природе
  Номер: 5
  Название: Земля и мы
  Год выпуска: 2014
Состояние единицы хранения:
  Инвентарный номер: 1235
  Наличие: False
^ labs/lab8/MyClass | main 0 ? >

```

Упражнение 3. Реализация прогрессии с помощью интерфейса.

В этом упражнении я заменил абстрактный класс `Progression` на интерфейс `IProgression`, определяющий поведение классов `ArithmeticProgression` и `GeometricProgression`, описывающих арифметическую и геометрическую прогрессии.

```
using System;
```

```
interface IProgression
{
    int GetElement(int k);
    int Sum(int n);
}
```

```
using System;

class ArithmeticProgression : IProgression
{
    private int a0;
    private int b;

    public ArithmeticProgression(int a0, int b)
    {
        this.a0 = a0;
        this.b = b;
    }

    public int GetElement(int k)
    {
        return a0 + (k - 1) * b;
    }

    public int Sum(int n)
    {
        return (2 * a0 + (n - 1) * b) * n / 2;
    }
}
```

```
using System;

class GeometricProgression : IProgression
{
    private int p0;
    private int q;

    public GeometricProgression(int p0, int q)
    {
        this.p0 = p0;
    }
}
```

```

        this.q = q;
    }

    public int GetElement(int k)
    {
        return p0 * (int)Math.Pow(q, k - 1);
    }

    public int Sum(int n)
    {
        if (q == 1) return p0 * n;
        return p0 * (int)((Math.Pow(q, n) - 1) / (q - 1));
    }
}

```

Тесты:

```

public static void Main(string[] args)
{
    IProgression arith = new ArithmeticProgression(2, 3);
    IProgression geom = new GeometricProgression(2, 2);

    Console.WriteLine("Арифметическая прогрессия:");
    Console.WriteLine("5-й элемент = " + arith.GetElement(5));
    Console.WriteLine("Сумма 5 элементов = " + arith.Sum(5));

    Console.WriteLine("\nГеометрическая прогрессия:");
    Console.WriteLine("5-й элемент = " + geom.GetElement(5));
    Console.WriteLine("Сумма 5 элементов = " + geom.Sum(5));
}

```

Пример:

```
^ labs/lab8/Progrssion | main @ ? >
dotnet run
Арифметическая прогрессия:
5-й элемент = 14
Сумма 5 элементов = 40

Геометрическая прогрессия:
5-й элемент = 32
Сумма 5 элементов = 62
```

Выводы.

В ходе проделанной лабораторной работы я использовал интерфейсы при реализации иерархии классов как важного элемента объектно-ориентированного программирования и приобрел навыки реализации интерфейсов.

Code review. (by zzzcode.ai)

Резюме

Класс `Book` представляет собой реализацию книги с различными атрибутами, такими как автор, название, издатель и т.д. Он наследуется от абстрактного класса `Item`, который управляет состоянием доступности книги. Код в целом хорошо структурирован, но требует некоторых улучшений в области стиля и производительности.

Ошибка

В методе `Return()` класса `Book` переменная `taken` устанавливается в значение `returnSrok`, которое по умолчанию равно `false`. Это может привести к тому, что книга будет считаться возвращенной, даже если она не была возвращена. Следует добавить логику для проверки состояния возврата.

Стиль кода

Использование `String` вместо `string` не соответствует общепринятому стилю C#. Рекомендуется использовать `string` (с маленькой буквы). Имена переменных и методов должны следовать

соглашениям о наименовании. Например, `returnSrok` можно переименовать в `isReturned` для большей ясности.

Структура кода

Класс `Book` имеет хорошую структуру, однако: Конструкторы перегружены, что может усложнить создание объектов. Рекомендуется использовать один конструктор с параметрами по умолчанию. Метод `SetBook` можно заменить на использование свойств, что улучшит инкапсуляцию.

Читаемость

Код читаем, но можно улучшить: Добавить XML-комментарии к методам и классам для лучшего понимания их назначения. Разделить длинные строки на несколько строк для повышения читаемости.

Производительность

Статическая переменная `price` может быть изменена через метод `SetPrice`, что может привести к неожиданным изменениям. Рекомендуется сделать её свойством только для чтения или использовать паттерн Singleton для управления ценой.

Масштабируемость

Класс `Book` может быть расширен для поддержки дополнительных атрибутов, таких как жанр или ISBN. Однако, если количество атрибутов увеличится, стоит рассмотреть возможность использования паттерна проектирования, такого как `Builder`.

Безопасность

Необходимо добавить валидацию входных данных в конструкторах и методах, чтобы предотвратить создание объектов с некорректными значениями (например, отрицательное количество страниц или год).

Обработка ошибок

В текущем коде отсутствует обработка исключений. Рекомендуется добавить обработку ошибок, особенно в методах, которые могут вызывать исключения, таких как `PriceBook`.

Заключение

Класс `Book` и его реализация в целом являются хорошей основой для работы с книгами. Однако, для улучшения качества кода и его поддержки, необходимо внести изменения в стиль, структуру и обработку ошибок. Рекомендуется также рассмотреть возможность добавления тестов для проверки функциональности классов.